AMSAA

TECHNICAL REPORT NO. TR-638

ARMY STANDARD UNIT OBJECT

DECEMBER 1998

U.S. ARMY MATERIEL SYSTEMS ANALYSIS ACTIVITY
ABERDEEN PROVING GROUND, MARYLAND 21005-5071

DESTRUCTION NOTICE

Destroy by any method that will prevent disclosure of contents or reconstruction of the document.

DISCLAIMER

The findings in this report are not to be construed as an official Department of the Army position unless so specified by other official documentation.

WARNING

Information and data contained in this document are based on the input available at the time of preparation.

TRADE NAMES

The use of trade names in this report does not constitute an official endorsement or approval of the use of such commercial hardware or software. The report may not be cited for purposes of advertisement.

AD NUMBER	DATE	DTIC ACCESSION NOTICE
1. REPORT IDENTIFYING INFORMATION		REQUESTER:
A. ORIGINATING AGENCY U.S. Army Materiel Systems Analysis Activity		1. *Put your mailing address on reverse of form.* 2. *Complete items 1 and 2.* 3. *Attach form to reports mailed to DTIC.* 4. *Use unclassified information only.*
B. REPORT TITLE AND/OR NUMBER TR-638		
C. MONITOR REPORT NUMBER		
D. PREPARED UNDER CONTRACT NUMBER		DTIC:
2. DISTRIBUTION STATEMENT Approved for public release; distribution is unlimited.		1. *Assign AD Number.* 2. *Return to requester.*

DTIC FORM 50 DEC 80 PREVIOUS EDITIONS ARE OBSOLETE

DEFENSE TECHNICAL INFORMATION CENTER
CAMERON STATION
ALEXANDRIA, VIRGINIA 22314

OFFICIAL BUSINESS
PENALTY FOR PRIVATE USE, $300

POSTAGE AND FEES PAID
DEFENSE LOGISTICS AGENCY
DOD-304

Director
U.S. Army Materiel Systems Analysis Activity
ATTN: AMXSY-[redacted] (392 Hopkins Rd) (RPM)
Aberdeen Proving Ground, MD 21005-5071

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (LEAVE BLANK)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	December 1998	Technical Report

4. TITLE AND SUBTITLE
Army Standard Unit Object

5. FUNDING NUMBERS

6. AUTHOR(S)
Don Hodge, Brad Bradley

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Director
U.S. Army Materiel Systems Analysis Activity
392 Hopkins Road
Aberdeen Proving Ground, MD 21005-5071

8. PERFORMING ORGANIZATION REPORT NUMBER
TR-638

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Director
U.S. Army Materiel Systems Analysis Activity
392 Hopkins Road
Aberdeen Proving Ground, MD 21005-5071

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION IS UNLIMITED.

12b. DISTRIBUTION CODE
A

13. ABSTRACT (Maximum 200 words)

Object-oriented programming offers the potential for increased code reuse, maintainability, and ease of developing simulations. Because of these benefits, the use of object-oriented technologies will increase over time. In order to prevent duplication of effort and the development of incompatible models, the Deputy Undersecretary of the Army for Operations Research (DUSA-OR) directed the development of an Army object management initiative to provide a foundation for Army object development. This report documents the standard Unit Object that defines the minimum set of objects and object methods needed for the development of Unit Objects in models and simulation.

14. SUBJECT TERMS
object oriented programming; modeling and simulation, unit representation

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAME AS REPORT

This page intentionally left blank.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS.

The U.S. Army Materiel Systems Analysis Activity (AMSAA) wishes to recognize the following individuals for their contributions to this report:

Authors:	Don Hodge Brad Bradley
Contributions:	Major Leroy Jackson
Technical Reviewers:	Charles E. Abel Wilbert J. Brooks

ACRONYM LIST.

AMSAA	Army Materiel Systems Analysis Activity
AMSMPWG	Army Modeling and Simulation Management Program Working Group
ARES	Advanced Regional Exploratory System
ASTARS	Army Standards Repository System
AWARS	Army Warfare Simulation
CAA	Concepts Analysis Agency
CASTFOREM	Combined Arms Support Task Force Evaluation Model
CSS	Combat Service Support
DNBI	Disease and Non-Battle Injuries
DUSA(OR)	Deputy Undersecretary of the Army for Operations Research
ITEM	Integrated Theater Engagement Model
IUD	Initial Unit Design
JWARS	Joint Warfare Simulation
M&S	Models and Simulations
ModSAF	Modular Semi-Automated Forces
NSC	National Simulation Center
OMSC	Object Management Standards Category
OMWG	Object Management Working Group
OOP	Object Oriented Programming
ROM	Refuel On the Move
SAMSO	Standard Army Modeling and Simulation Object
SNAP	Standards Nomination and Approval Process
STRICOM	Simulation, Training, and Instrumentation Command
TRAC-FLVN	TRADOC Analysis Center Ft. Leavenworth
TRAC-MTRY	TRADOC Analysis Center - Monterey
TRAC-WSMR	TRADOC Analysis Center - White Sands Missile Range
TRADOC	Training and Doctrine Command
VIC	Vector-In-Commander
WARSIM	Warfighter Simulation

ARMY STANDARD UNIT OBJECT

1. INTRODUCTION

This report documents the development of the Army standard Unit Object. For this effort, the definition of a Unit encompasses military organizations that represent collections of entities (e.g., people, vehicles, weapon systems, etc.). Examples of this definition include organizations (i.e., companies, battalions, brigades, divisions, etc.) as well as functional groups (e.g., Tactical Operations Centers and Fire Control Centers). These types of groups are typically used in simulations where the interest is in representing the sum or aggregate performance and/or behavior of the group versus representing the performance, behavior or characteristics of the individual elements that compose the group. Simulations that typically exercise this structure are known as "aggregate-level simulations."

2. BACKGROUND

Many of the current Army and Joint model development efforts have embraced the use of Object Oriented Programming (OOP) for their model development efforts. As a result, there has been a proliferation of competing object models. In 1QFY97, the Deputy Undersecretary of the Army for Operations Research (DUSA(OR)) formed an Object Management Working Group (OMWG) to propose a policy addressing the need for standards associated with Army M&S objects. The proposed policy developed by the OMWG recommended that the Army focus on a high-level object class structure, independent of any specific simulation environment. This would allow M&S developers to tailor the high-level object standards to their specific applications through lower-level class/ instantiations that extend the standards to a specific M&S requirement. The overall impact in the development of standard abstract objects will be to organize future M&S along a common object structure to support interoperability, object reuse, and community understanding of the M&S. The proposed policy was briefed by the OMWG to the DUSA(OR) and was accepted in principle. AMSO subsequently formed the Object Management Standards Category (OMSC) in April 1997 to initiate the proposed policy. The OMSC mission is to:

- develop abstract objects for Army M&S functions,
- identify the minimum set of object methods/public data associated with the object function, and
- link the object methods to standard algorithms/data sources obtained from the other AMSO standard categories.

The OMSC is comprised of M&S practitioners to include those from the following agencies:

- Army Materiel Systems Analysis Activity (AMSAA) -- serves as the OMSC Coordinator;
- Concepts Analysis Agency (CAA);
- National Simulation Center (NSC);
- TRADOC Analysis Center - Ft. Leavenworth (TRAC-FLVN);
- TRAC- Monterey (TRAC-MTRY),
- TRAC-White Sands Missile Range (TRAC-WSMR); and
- Simulation, Training, and Instrumentation Command (STRICOM).

3. APPROACH

During the initial stages of developing a policy on objects, AMSO funded the U.S. Army Training and Doctrine Command (TRADOC) Analysis Center, Monterey, California (TRAC-MTRY) to perform the "Standard Army Modeling and Simulation Object (SAMSO) Study"[1]. The study proposed an object development approach based on object composition. The OMSC reviewed the SAMSO approach and adopted it for use in developing Army Standard objects. A paper describing the component approach to model development is provided in Appendix A.

As a part of the SAMSO study, the study team developed sample Platform and Unit Objects. The OMSC selected the sample Unit Object design as the initial prototype for developing a standard Army Unit Object. To explore the capability of the Unit Object to address expected M&S unit implementations, the OMSC conducted a test application. The simulation chosen for the test application was the Army Warfare Simulation (AWARS). The results of this test application were used to refine the Unit Object. Additionally, to gain a broader perspective on the application of the draft Unit Object to other M&S domains, the revised draft Unit Object was provided to the Army M&S Management Program Working Group (AMSMP WG)[2] and to the Army M&S Standard Categories for review. Comments were collected and reviewed to determine if any changes to the Unit Object were needed to address M&S requirements. Based on these reviews, an updated version of the draft Unit Object was developed and submitted to the Standards Nomination and Approval Process (SNAP) and to the Army Standards Repository System (ASTARS).

[1] Buss, Arnold, and Leroy Jackson (September 1997), "Standard Army Modeling and Simulation Objects: Interim Report", US Army TRADOC Analysis Center – Monterey.

[2] Renamed as the Policy & Technology Working Group

4. INITIAL DESIGN

An output of the SAMSO Study was a draft Platform Object and Unit Object. (The results of the work on the Platform Object are described in AMSAA TECHNICAL REPORT NO. TR-634). Members of the SAMSO study team reviewed documentation from a number of existing and developing Army models. The models reviewed included: Eagle; Integrated Theater Engagement Model (ITEM); Joint Warfare Simulation (JWARS); Modular Semi-Automated Forces (ModSAF); and Warfighter Simulation (WARSIM) 2000. Based on this research, the study team identified a set of components that were common to the units represented in the models.[3] This Initial Unit Design (IUD) is shown in Figure 1.

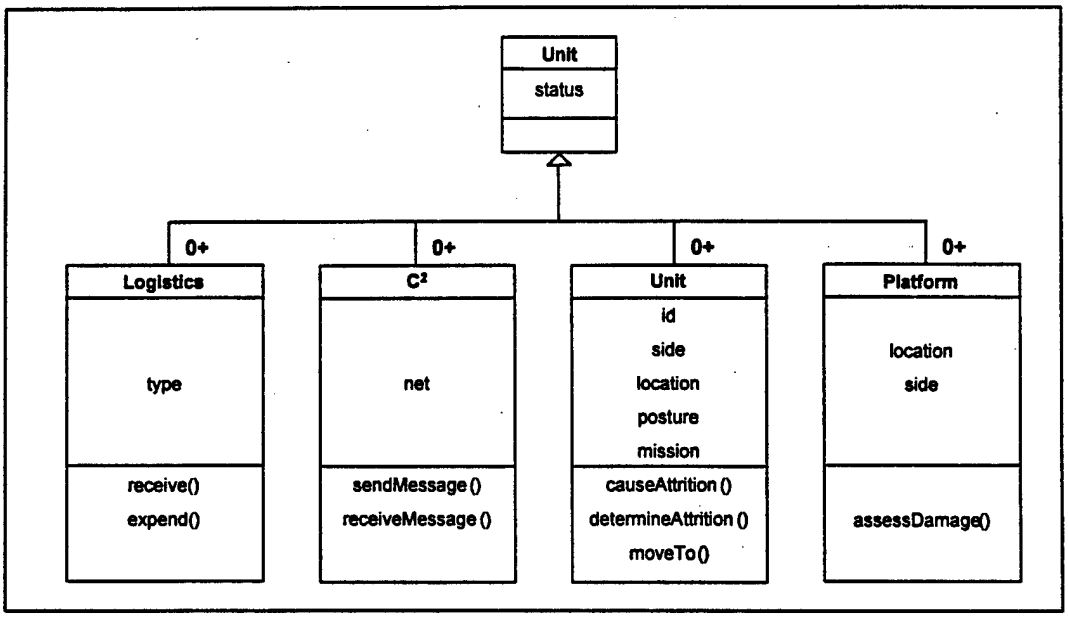


Figure 1. Initial Unit Object Design.

[3] Cotton, Arthur L. III (September 1997) "Development of Standard Unit-Level Army Object Model", MS Thesis, Department of Operations Research, Naval Postgraduate School.

5. TEST DESIGN - AWARS UNIT OBJECT

The basic philosophy behind the development of any standard object is its use as a building block in the development of model-specific objects. In order to determine the utility of the proposed standard Unit Object, the IUD was used to develop sample Unit Objects compatible with an aggregate-type model. The model used to test the IUD was the Army Warfare Simulation under development at TRAC-FLVN.

AWARS is a corps/division-level, low-resolution, command-and-control model. As a corps-level model, AWARS will integrate units from battalion through corps to represent a combined-arms battlefield. AWARS will replace both the Eagle and Vector-In-Commander (VIC) models. The AWARS model uses an object oriented design that shares many elements in common with the Eagle model.[4]

On November 12-14, 1997, the SAMSO study director (Major Jackson), members of the OMSC (Don Hodge) and the AWARS design team (Mike Hannon, Terry Gach, Mike Fraka) met to apply the IUD to the development of an AWARS-compatible Unit Object. The resulting object was composed of eleven components, with five coming from the IUD. Figure 2 shows the composition of the resulting AWARS-compatible Unit Object design.

An assessment of the utility of the IUD to support development of AWARS-compatible unit objects identified a number of issues. On the one hand, all of the IUD object components were used in the AWARS Unit Object with little or no modification. As would be expected, the AWARS Unit Object design did contain model-specific additions to the IUD, but the number of these additions was small. The small number can be attributed to the fact that AWARS is based on an object oriented design and, as the model is still under development, the design team focused on the required generic elements versus capturing implementation-specific details.

[4] The Eagle model was developed from an object oriented design and was written in the Common Lisp Object System (CLOS) language.

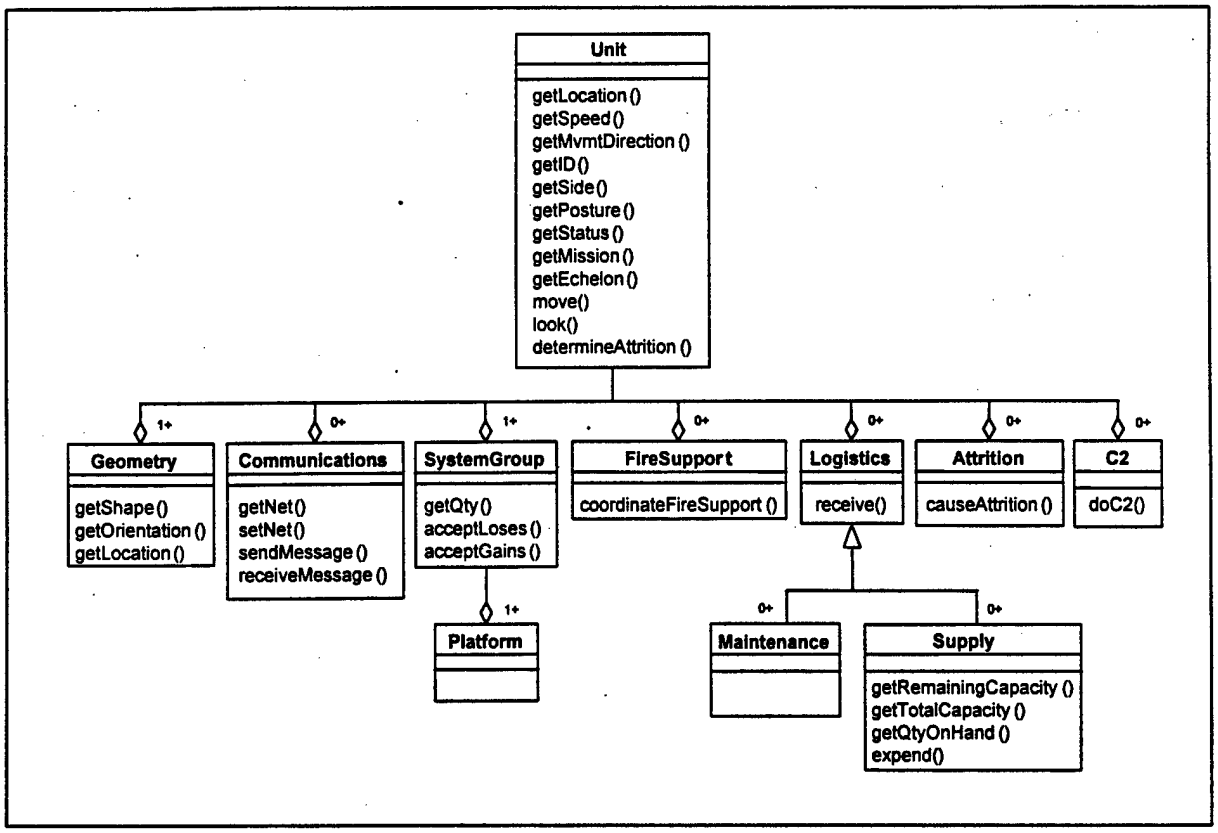


Figure 2. AWARS Unit Object Design.

On the other hand, there were several additional object components added to the IUD that represent functions that are generic in nature and would be required for aggregate-type Unit Objects. One of these additional components addresses the requirement to provide a description of the physical characteristics and composition of aggregate-type units. Specifically, all aggregate-type units occupy and/or are responsible for a given amount of terrain. Some method needs to be available to represent this footprint (i.e., area and orientation). Additionally, aggregate units are, by definition, composed of a number of individual systems. The number and type of each system figure into the attrition, mobility, and logistics calculations. Another area relates to the concept of command and control. Again, by their nature, aggregate units need to represent the ability to command and control subordinate units. A third area is reflected in the AWARS component "ATTRITION." Current Army-accepted attrition methodologies differ according to the damage-causing mechanism. Unlike entity-type objects that contain only one type of damage-causing mechanism (i.e., direct

fire versus indirect fire), aggregate units can contain both. This multiple nature needs to be captured within the object framework. A fourth area relates to where the model cognitive/ decision making processes should reside. In most simulations, there are identified decisions and/or choices that are required as a simulation executes. For aggregate-type units, there are both behaviors (e.g., how does a battalion conduct a hasty attack) as well as cognitive/decision making/planning processes. The IUD structure, as used during these sample object development efforts, did not contain a clear location or component to host these types of behavior/cognitive processes.

6. UNIT OBJECT DESIGN REVIEW

After the test application using the AWARS simulation, the OMSC met to agree on required modifications to the draft Unit Object. In addition, the modified draft design for the Unit Object was provided to a number of groups throughout the Army for review and comment. These groups included the Army Model and Simulation Management Program Working Group (now the Policy and Technology WG) and the Army Model and Simulation Standards Category Committees. The review results included written input from the WARSIM simulation developers and the logistics community. The results of the OMSC review along with a summary of the other comments are provided in this section.

6.1 OMSC Review. On November 18-19, 1997, the OMSC met to review the results of the Unit Object design test efforts. The members present for this meeting were Brad Bradley (Chairman), Don Hodge (AMSAA), John Shepherd (CAA), Sean Mackinnon and Kevin Gipson (NSC), Mike Hannon and Terry Gach (TRAC-FLVN), Major Jack Jackson (TRAC-MTRY), Carol Denney and Donna Vargas (TRAC-WSMR), and Ben Paz (STRICOM). After the review of Unit Object design test efforts, the OMSC modified the IUD in the following ways:

1. Added a new component (UnitGeometry) to provide a description of the geometry of the unit on the ground,
2. Added a new component (SystemGroup) to provide a description of the number and types of systems owned by the unit,
3. Added a new sub-component (PlatformInformation) to provide a location for system characteristics data,
4. Moved the Platform component from directly supporting the Unit Object to supporting the SystemGroup component,
5. Added a new component (C2) to provide a location to place Command-and-Control functions,
6. Changed the attribute data found in the IUD to methods that would return the attribute data, and
7. Added a number of new methods to the existing components. (e.g. getNet(), setNet(),look(), getEchelon(), getTotalCapacity(), etc.)

The interim design is shown in Figure 3.

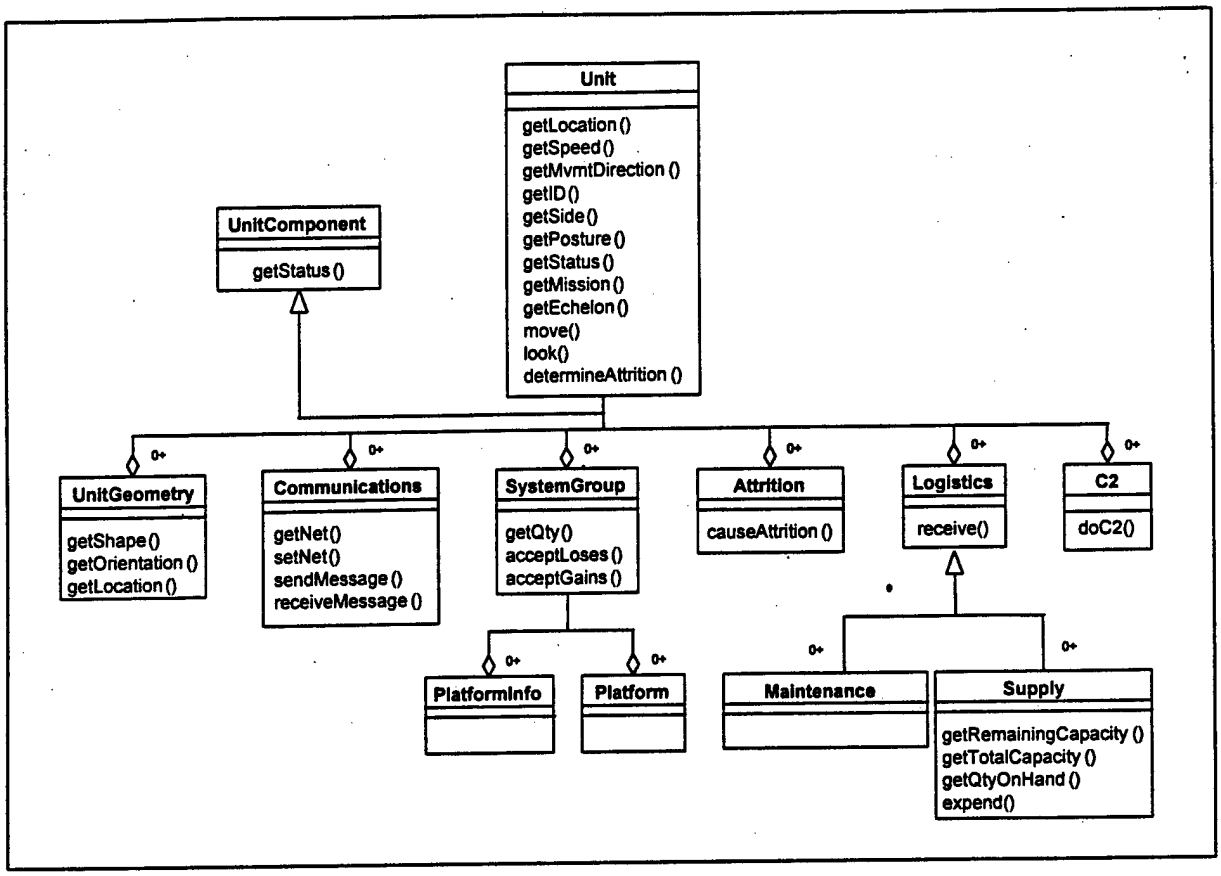


Figure 3. OMSC Interim Unit Object Design.

6.2 ARES Review. ARES is a multi-resolution, Joint-force, theater-level model. ARES is being developed by the General Research Corporation for the Concepts Analysis Agency to address a broad spectrum of regional conflicts. As a multi-resolution model ARES can portray a wide spectrum of modeling entities ranging from a single sensor system to an Army Corps containing thousands of systems. ARES is based on an object oriented design. The interim Unit Object design was reviewed by both the government and contractor ARES design teams. Their assessment was that the proposed interim Unit Object design could have been used to build ARES. They made no recommended changes based on this review.

6.3 WARSIM Review. Representatives from the National Simulation Center (Sean MacKinnon and Kevin Gipson) did a comparison between the interim Platform and Unit Objects and similar objects being developed for the WARSIM 2000 program (Appendix B). In this review, the authors identified major differences in the organization and structure of the two Unit Objects. In the WARSIM design, the functional elements of the OMSC Unit Object were partitioned between three different objects. Figures 4-6 show the WARSIM objects. This difference is attributable to the different assumptions made in developing each design. The WARSIM 2000 design mirrors the Operational Requirements Document developed for the WARSIM 2000 program. The interim standard Unit Object is oriented around physical processes and functions.

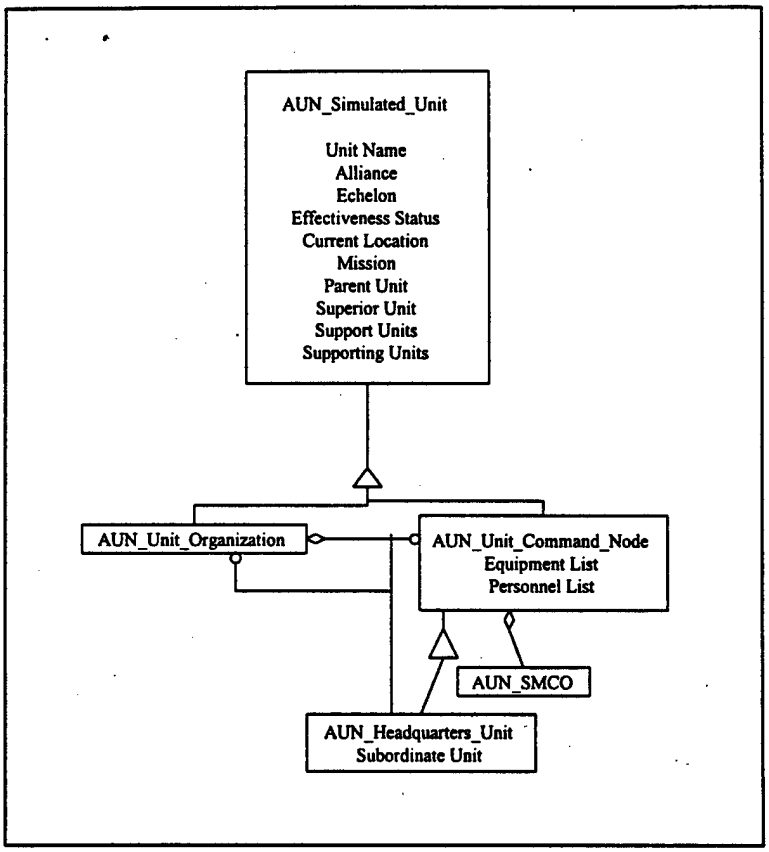


Figure 4. WARSIM Unit Model.

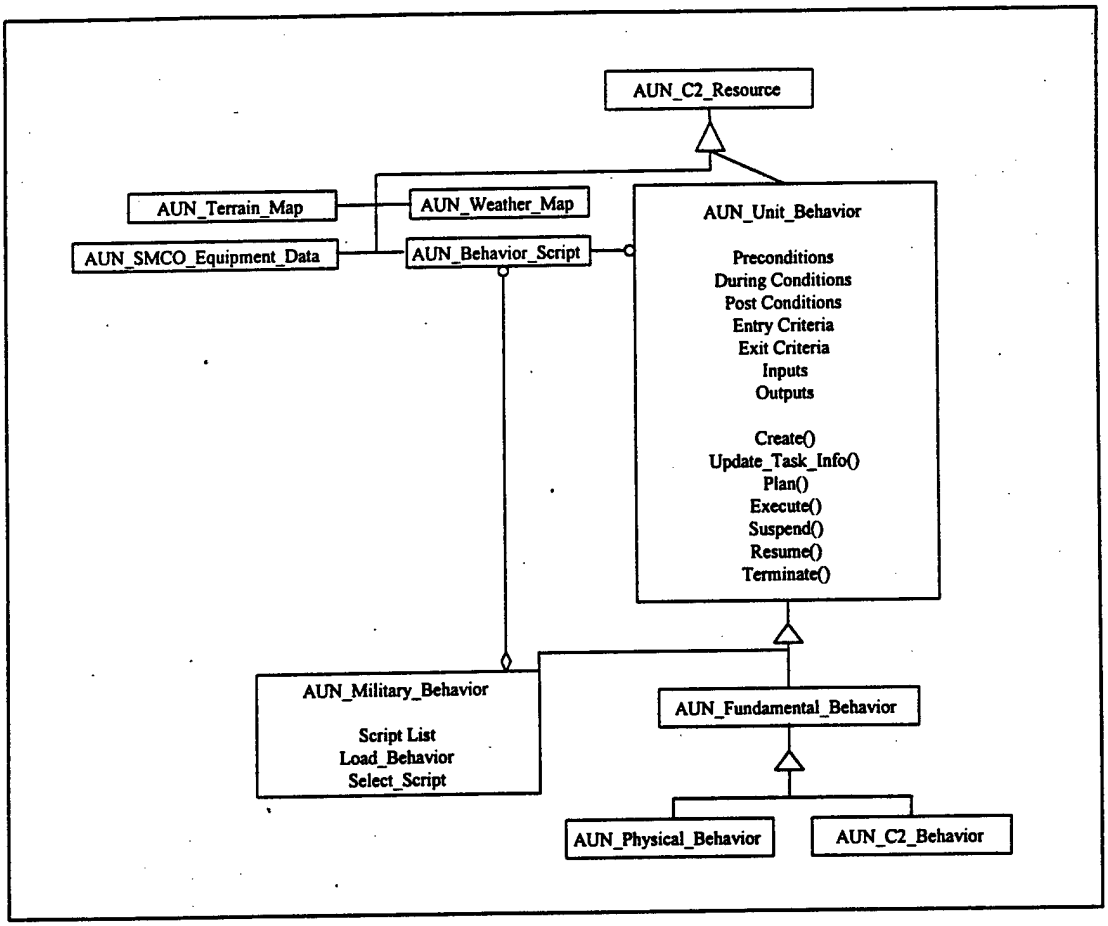


Figure 5. WARSIM AUN_C2_Resource Object Design.

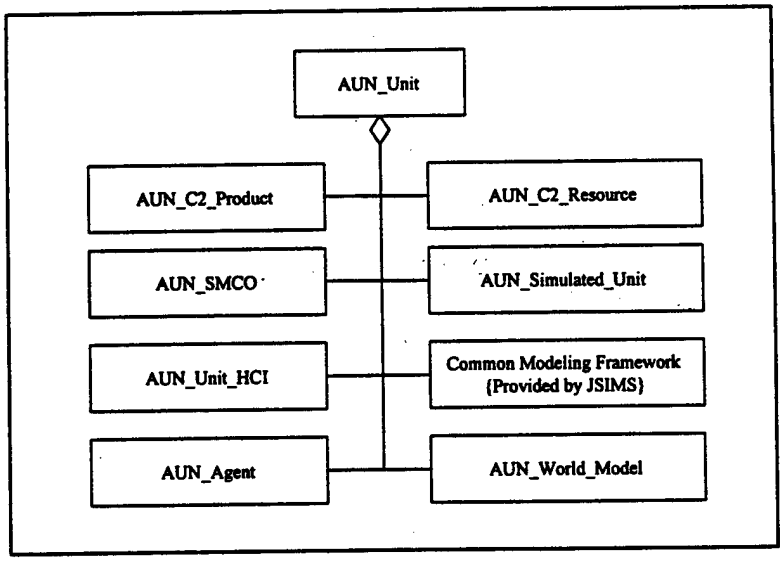


Figure 6. WARSIM AUN_Unit Object Design.

Table 1 provides a comparison between the functions performed by the components of each design. From this table, we can see that the functions identified in the OMSC Unit Object are contained in the WARSIM design. The differences between the two designs relate to the location of some of the functions and the nomenclature used to describe some of the functions. Based on this review, no changes were made to the interim Unit Object definition.

Table 1. Comparison of OMSC and WARSIM 2000 Functional Components.

OMSC	WARSIM
Unit	**AUN_Simulated_Unit**
GetID()	Unit Name
GetSide()	Alliance
GetEchelon()	Echelon
GetStatus()	Effectiveness Status
GetLocation()	Current Location
GetMission()	Mission
GetSpeed() GetMvmtDirection() GetPosture() DetermineAction()	**AUN_C2_Behavior** (see Figure 4 for details about organization)
Move()	**AUN_Physical_Behavior** (see Figure 4 for details about organization)
Datalook()	**AUN_SMCO_Equipment_Data** passes info to **AUN_SMCO**

Table 1. Comparison of OMSC and WARSIM 2000 Functional Components. (Continued)

OMSC	WARSIM
SystemGroup GetQty() AcceptLoses() AcceptGains()	**AUN_Unit_Command_Node**
Platform	**AUN_SMCO**
Geometry GetShape() GetOrientation() GetLocation()	**AUN_C2_Behavior**
C2 DoC2()	**AUN_C2_Resource**
Attrition CauseAttrition()	**AEQ_Equipment** sends info to **AUN_SMCO_Equipment_Data**
Logistics Receive()	**AEQ_Equipment**
Maintenance	**AEQ_Equipment**
Supply GetRemainingCapacity() GetTotalCapacity() GetQtyOnHand() Expend()	**AEQ_Equipment**
Communications GetNet() SetNet() SendMessage() ReceiveMessage()	**AUN_SMCO**

6.4 Combat Service Support (CSS). As a result of discussions between the OMSC and Logistics SC members at the May 1998 Army M&S Standards Workshop, the OMSC was provided a list of the minimum CSS requirements to be represented in combat simulations. The list is comprised of the following sets:

ARM
- Conduct ammo transfer operations.
- Account for direct and indirect fire ammo by type.

FUEL
- Conduct fuel transfer operations, including Refuel On the Move (ROM).
- Provide visibility of fuel quantities on hand.

MAN & MEDICAL
- Conduct medical evacuation and treatment operations.
- Generate types of combat and Disease and Non-Battle Injuries (DNBI) casualties.

FIX
- Conduct maintenance operations.
- Conduct evacuation and recovery operations.
- Generate combat and reliability failures.

After reviewing these requirements and the interim Unit Object design, the OMSC addressed each as follows:

- The Supply Sub-Component of the Logistics Component of the interim Unit Object addresses the following CSS elements:
  - ARM - Account for direct and indirect fire ammo by type.
  - FUEL - Provide visibility of fuel quantities on hand.

- Addition of the method "transfer()" to the Supply Sub-Component of the interim Unit Object will address the following CSS elements:
  - ARM - Conduct ammo transfer operations.
  - FUEL - Conduct fuel transfer operations, including ROM.

- Add the method "conductMaintenance" to the Maintenance Sub-Component of the Logistics Component of the interim Unit Object to address the following CSS elements:
  - MAN & MEDICAL - Conduct medical treatment operations.
  - FIX - Conduct maintenance operations.

- Add the method "conductRecovery" and "conductEvacuation" to the Maintenance Sub-Component of the Logistics Component of the interim Unit Object to address the following CSS elements:

  - MAN & MEDICAL - Conduct medical evacuation operations.
  - FIX - Conduct evacuation/recovery operations.

- Generation of combat casualties and combat damage should be addressed by the appropriate methodologies in the determineAttrition() method of the interim Unit Object.

7. FINAL UNIT OBJECT DESIGN AND DEFINITIONS

7.1 Final Unit Object Design. Figure 7 shows the final design for the Unit Object. This design is based on the OMSC review documented in this report and input provided by the M&S community. This design was nominated in the Standards Nomination and Approval Process for placement into the Army Standard Repository System.

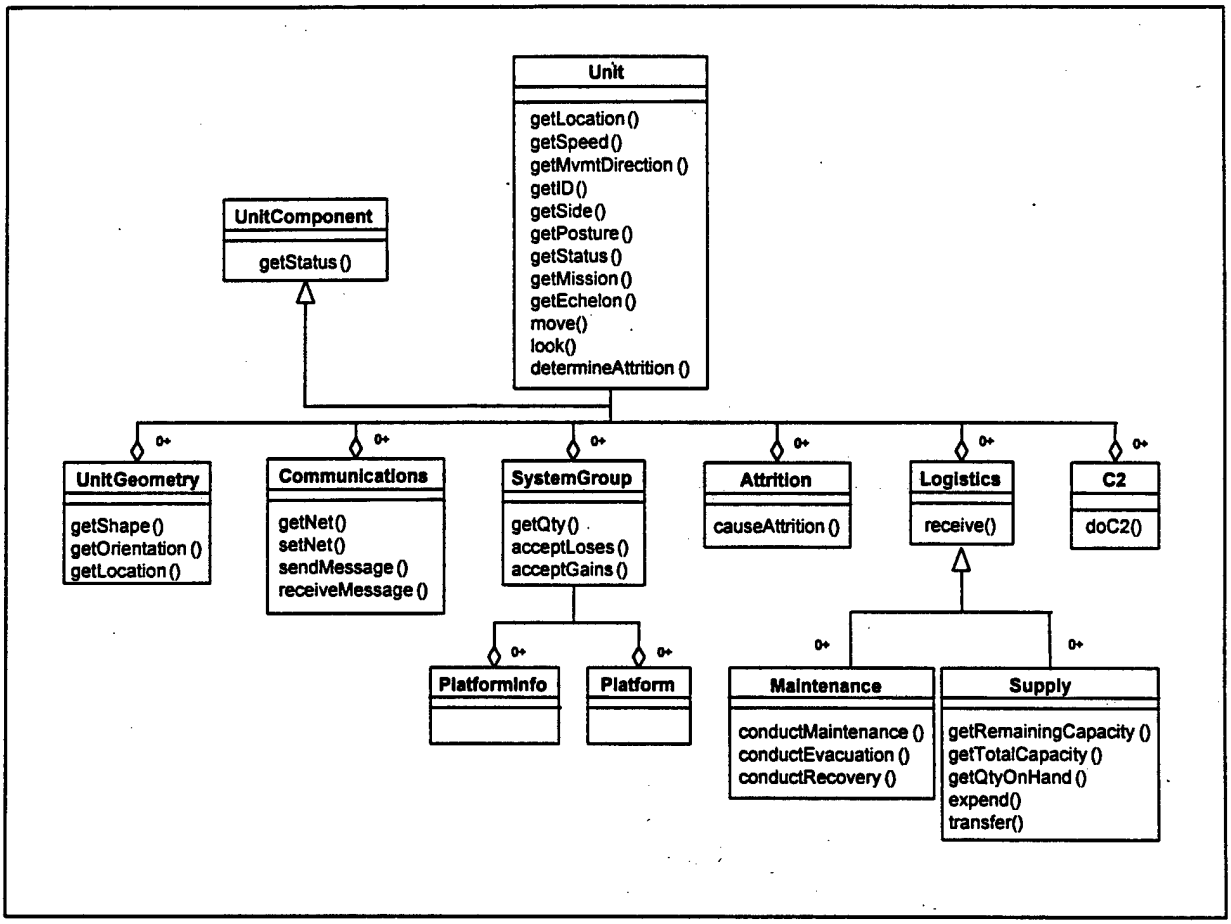


Figure 7. OMSC Final Unit Object Design.

7.2 Unit Object Class And Component Definitions. A detailed description for each of the components and methods contained in the Unit Object standard definition is provided below.

Class Unit: A "Unit" is any military organization that is composed of multiple entities. Examples include military organizations such as a company, battalion, brigade, or division

Public Methods:

getLocation(): Returns the current unit location. Typically this is the center of mass or some other point location representative of the unit location.
getSpeed(): Returns the current movement speed for a unit that is moving from one location to another.
getMvmtDirection(): Returns the movement direction for a unit moving from one location to another.
getID(): Returns a string that identifies the unit.
getSide(): Returns the faction or coalition for the platform. There is no implied enmity between sides.
getPosture(): Returns the unit posture. Examples of posture might be operational activities like road march, hasty attack, hasty defense, etc.
getStatus(): Returns the unit status. Status is used for planning. Examples might include a percent effectiveness (based on system weights), fraction on hand (number on hand divided by number authorized), unit effectiveness state (an enumerated type based on the percent effectiveness), relationship with objective (an enumerated type based on distance to current objective). There may also be a status for fuel and weapons and a status based on enemy fire.
getMission(): Returns the unit mission. An example is the current task the unit was ordered to accomplish.
getEchelon (): Returns the unit echelon. Examples are battalion, brigade and division.
move(): Used to advances a unit toward its next location.
look(): Used to initiate local detection using the unit search capabilities (probably used if the unit has sensors or is in contact with an enemy unit).
determineAttrition(): Used to calculate the attrition caused by another unit or platform.

Class UnitComponent: A "Unit" is partitioned into logical components so that the modeler can compose a unit from various components. Components may be extended through inheritance. All of the components listed below will inherit the following method from this class.

Public Methods:

getStatus(): Returns the status of the unit or component.

Class UnitGeometry. The unit geometry describes the shape or footprint of the unit on the ground, the layout of systems within the unit, the unit search area, and unit orientation and posture. Geometry may be used for attrition, sensing and movement.

Public Methods:

getShape(): Returns the bounding shape of the unit.

getOrientation(): Returns the general orientation of the systems within the unit location.

Class SystemGroup. This component accounts for individual systems (or platforms) within the unit.

Public Methods:

getQty (): Return the number of systems of this type in the unit.

acceptLosses (): Used to decrement the number of systems of this type in the unit.

acceptGains (): Used to increment the number of systems of this type in the unit.

Class Platform. A platform can be any entity of interest in the model. Examples include vehicles of all types, individuals/ persons, individual systems (i.e., radar systems), a missile, etc. The complete definition for this class is provided in a separate section.

Class PlatformInfo: This component contains static information and/or data about the various platforms contained within the unit. Examples include the gross weight of a vehicle, a description of the size or type of weapons mounted on the platform, etc.

Class Logistics. This component is intended to capture or represent the internal logistics capability and/or requirements of the unit. This covers both supply and maintenance requirements and/or activities.

Public Methods:

getType(): Returns the type of logistics (supply or maintenance).

receive(): Used to increment the quantity of this logistic component.

Class Supply. A supply component of a unit such as ammunition class.
Derived from Logistics

Public Methods:

getRemainingCapacity(): Returns the remaining capacity for this supply component.

getTotalCapacity(): Returns the total capacity for this supply component.

transfer(): Used to transfer a quantity of an on hand supply component to another unit or platform.

Class Maintenance: A maintenance component of a unit such as a repair action.
Derived from Logistics

Public Methods:

conductMaintenance(): Used to perform maintenance actions on equipment and medical treatment for individuals.

conductRecovery(): Used to recover items from an area of operations.

conductEvacuation(): Used to evacuate equipment and/or individuals to rear areas.

Class C2. This component is used for command and control decision making in the unit. A unit may have more than one command and control component (for itself, for subordinate units, and for other units).

Public Methods:

doC2(): Used to initiate a command and control cycle where command decisions are made and control actions initiated.

Class Attrition. The attrition component allows the unit to cause losses to another unit. This is shown as a separate class because a unit can have more than one way to inflict damage on another unit (i.e., direct fire systems, indirect fire systems, etc).

Public Methods:

CauseAttrition(): Used for the unit to cause losses to another unit.

Class Communications: This component provides the ability to explicitly model communications.

Public Methods:

getNet(): Returns the collection of objects capable of exchanging messages.
setNet(): Used to add the unit to the collection of objects capable of exchanging messages.
sendMessage(): Used to send a message on the net.
receiveMessage(): Used to receive a message from the net.

APPENDIX A - A COMPONENT APPROACH TO OBJECT MODEL STANDARDS FOR SIMULATION

THIS PAGE INTENTIONALLY LEFT BLANK.

A Component Approach to Object Model Standards for Simulation

Major Leroy A. Jackson
Operations Research Analyst
U.S. Army TRADOC Analysis Center—Monterey
(408) 656-4061
jacksonl@mtry.trac.nps.navy.mil

Summary. Object models are an important feature of the United States Department of Defense (DoD) High Level Architecture (HLA) and the Defense Modeling and Simulation Office (DMSO) Conceptual Model of the Mission Space (CMMS). Currently, all major DoD simulations under development use object-oriented methodologies. The major benefits of object-oriented programming include software reuse, improved maintainability, interoperability, and rapid prototyping. A set of standard objects is needed to establish consistency among future Army models and simulations. This paper describes a component approach proposed for object model standards development.

1. INTRODUCTION

This paper describes a component approach for object-oriented modeling and design which has been adopted for standards development in the U.S. Army modeling and simulation community. This design approach directly supports the goals for developing object modeling standards by fostering model reuse and improving model interoperability.

2. BACKGROUND

In May 1997, the U.S. Army Training and Doctrine Command (TRADOC) Analysis Center (TRAC) in Monterey, California (TRAC—Monterey) began a study sponsored by the Army Modeling and Simulation Office (AMSO) to support standards development for Army modeling and simulation objects. [1] The study team was led by a military analyst at TRAC—Monterey and included a professor and two graduate students from the Operations Research Department of the Naval Postgraduate School. The study advisory group included senior analysts from the major Army analytical agencies. The team examined selected models from existing and future simulations under development in order to provide examples and insights to support object standards development. The team also developed an approach to object model standards development, drafted sample standards for platforms (entities) and units, and drafted sample guidelines for the use of standard objects. The study team determined that object model standards would focus on high-level abstract classes containing a minimal, essential set of class methods. Rather than specify standard attributes for classes, *get* and *set* methods would signify

the data content of standard objects. An important aspect of the study team recommendations was the component approach to object model standards.

3. APPROACHES TO REUSE

The two main approaches to reuse in object oriented designs are class inheritance and object composition. [2&3] Each approach has distinct advantages and disadvantages.

3.1 Inheritance

Inheritance allows subclasses to extend and specialize a parent class by adding data and methods, and by replacing the method implementation of the parent class with a new implementation. Inheritance is straightforward since it is directly supported by object oriented languages. General classes are placed higher in the inheritance hierarchy and more specialized objects lower, so several subclasses may reuse the parent class. Inheritance, however, breaks encapsulation by exposing the parent class implementation to its subclasses. Implementation changes in the parent class often necessitate changes in subclasses. Issues of multiple inheritance and the requirement for compile-time binding further dilute the value of inheritance for reuse. Inheritance promotes implementation dependencies. Despite some minor disadvantages, inheritance is an extremely important feature in object oriented systems. Inheritance of abstract classes provides common protocols or interfaces in an object-oriented design. This technique ameliorates some of the pitfalls in the use of inheritance.

3.2 Object Composition

Object composition is the construction of a class using instances of other classes as components. Because component classes are accessed through their interface (public methods), encapsulation is not broken and there are significantly fewer implementation dependencies. Object composition is, however, more difficult. It requires that component classes have well defined interfaces that promote reuse. In addition, objects must respect these interfaces since no implementation details are exposed. Finally, object composition proliferates numerous small component classes since each component class must focus on relatively few tasks. This often requires many interrelationships among the component classes that would normally be encapsulated in one larger class.

3.3 The Component Approach to Standards

The component approach to standards favors object composition over class inheritance, but exploits the advantages of both approaches. With the component approach, classes of interest are constructed by selecting and implementing abstract component classes. Component classes are implemented and possibly extended through inheritance. The principle advantage of the component approach to standards over alternative approaches is it focuses on the development of

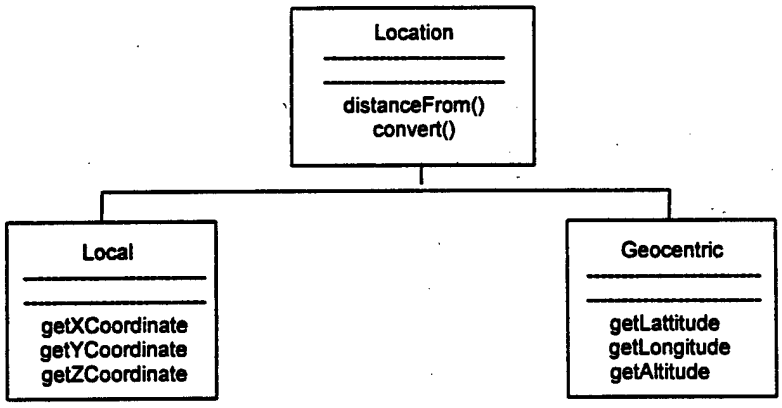
standard interfaces rather than the construction of a single monolithic class hierarchy. If a single class interface supports several different implementation schemes, then the goal of "plug and play" software components is achieved. For example, if the same method signature (set of parameters required to invoke the method) supports several attrition schemes (Lanchester, Bonder-Ferrel, etc.) then it is possible to substitute one attrition algorithm for another without making other changes in the simulation.

4. STANDARD M&S OBJECTS

This section provides examples of standard modeling and simulation (M&S) objects developed using the component approach and discusses the problem of determining the appropriate level of detail for standards using the component approach.

4.1 *Location* Class Example

The notion of location is fundamental to most military simulations. There are numerous coordinate systems used in simulation; each is appropriate for some simulations and not suitable for others. A common, abstract location object can foster interoperability among simulations that use different coordinate schemes. In this example (see next page), the *Location* class abstracts the concept of location by providing a method to calculate the distance between locations and to convert to an unspecified standard location scheme. The *Location* class has two standard subclasses, *Local* and *Geocentric*, which illustrate the two main competing coordinate schemes. Each provides location through *get* methods. [4] The *Location* class is powerful and flexible. Suppose one has a simulation that uses a network of arcs and nodes. The distance between nodes is stored in a table and the distance from a node along an arc is calculated based on the fraction of the arc traversed at the time a distance is requested. The simulation developer conforms to the standard by simply subclassing the *Location* class and implementing its methods.



Location Class Hierarchy

4.2 *PlatformComponent* Example

Entity level simulations of combat generally have a notion of platform or entity upon which most militarily significant actors from individual combatants to tanks to aircraft are based. While the details vary significantly among various simulations, there are common aspects of all platforms in almost all entity level simulations. The standard platform components are *Location*, *Communications*, *Movement*, *Sensor*, *Weapon*, *Carrier*, *Crew*, *PlatformFrame* and *Logistics* (with *Supply* and *Maintenance* subclasses). These components are subclasses of the *PlatformComponent* class that provides *getType* and *getStatus* methods to all components. (The interested reader can refer to [4, 5 and 9] for the details of the platform components.) A simulation developer composes platforms in an entity-level simulation using zero or more of each of components as appropriate. Implementation details are left to the developer, but each component provides a standard interface into a significant aspect of the entity as illustrated by the *Location* class described above. The standard platform components are flexible. The simulation developer uses only the components required in the simulation. If, for example, the crew is not modeled, then that component is omitted. There is no restriction on the number or type of weapons, sensors or communications systems on the platform.

4.3 Levels of Detail for Standards

The component approach does not solve the problem of determining the appropriate level of detail for standard classes, but it provides a suitable context for debate on this issue. The study team used several general rules to determine if a method belonged in a standard class. The primary rule was that the method be essential to support a function found in almost all simulations where the component would be found. The study team made a conscious effort to err on the side of proposing minimal standards to avoid creating a large burden for the simulation developer. The shared vision was of abstract components as the basis for standards. In the approach described, the abstract components are sufficient to assemble a platform that represents the abstract tank. Further refinement would be required to produce a generic tank and still more refinement to produce a detailed model of an actual tank. Each level is a possible standard, but the fraction of simulations which might support the more detailed standards is rather small.

5. CONCLUSION

The U.S. Army modeling and simulation community is reviewing standard component models for platform and unit objects which evolved from the study. The Object Management Standards Coordinating Committee has proposed a general framework for object model development and is actively developing standard component models for a variety of other significant objects found in ground combat simulations. The component approach to object modeling promotes reuse of models and improves model interoperability. It focuses on the development of a standard object interface which consists of the minimum, essential set of abstract class methods in a component.

6. ACKNOWLEDEMENTS

This work was sponsored by the Deputy Undersecretary of the Army for Operations Research through the auspices of the U.S. Army Modeling and Simulation Office. I am particularly indebted to Professor Arnold Buss of the Naval Postgraduate School for his keen insights and tremendous contributions to the study.

7. ABOUT THE AUTHOR

Major Leroy A. Jackson is an Army officer with over 20 years of enlisted and commissioned service. He graduated with a BA in Mathematics from Cameron University in 1990 and with an MS in Operations Research from the Naval Postgraduate School in 1995. He is currently an operations research analyst at the U.S. Army Training and Doctrine Command (TRADOC) Analysis Center (TRAC) Research Activities in Monterey, California and continues graduate studies in operations research at the Naval Postgraduate School.

8. REFERENCES

[1] Jackson, Leroy A. (April 1997) *Standard Army M&S Objects Study Plan*, US Army TRADOC Analysis Center—Monterey.
[2] Gamma, Erich, Richard Helm, Ralph Johnson and John Vlissides (1995), *Design Patterns: Elements of Reusable Object-Oriented Software Reuse*, Addison-Wesley.
[3] Jacobson, Ivar, Magnus Christerson, Patrik Jonsson and Gunnar Overgaard (1995), *Object-Oriented Software Engineering: A Use Case Driven Approach*, Addison-Wesley.
[4] Buss, Arnold, and Leroy Jackson (September 1997), *Standard Army Modeling and Simulation Objects: Interim Report*, US Army TRADOC Analysis Center— Monterey.
[5] Dudgeon, Douglas E. (September 1997) *Development a Standard Platform-Level Army Object Model*, MS Thesis, Department of Operations Research, Naval Postgraduate School.
[6] Cotton, Arthur L. III (September 1997) *Developing a Standard Unit-Level Object Model*, MS Thesis, Department of Operations Research, Naval Postgraduate School.

THIS PAGE INTENTIONALLY LEFT BLANK.

APPENDIX B - WARSIM 2000 CROSSWALK WITH THE OMSC OBJECT MODEL STANDARD

THIS PAGE INTENTIONALLY LEFT BLANK.

WARSIM 2000 Crosswalk with the OMSC Object Model Standard

26 Feb 98

Sean MacKinnon
National Simulation Center
(mackinns@leav-emh1.army.mil)

Kevin Gipson
National Simulation Center
(gipsonk@leav-emh1.army.mil)

Background

The OOA approach chosen by the WARSIM IDT closely follows the Rumbaugh OMT methodology. The WARSIM IDT extracted nouns and noun phrases from the Operation Requirements Document (ORD) to identify the object classes required within WARSIM and to establish traceability back to user requirements. A simplified model of this process is illustrated in Figure 1. This approach drove the IDT away from the development of a functionally oriented class structure, therefore, a lot of differences have been noted between the two unit models. As an example, the WARSIM unit model does not contain functional classes such as Attrition, Geometry, Logistics, etc. Because of the fundamentally different OOA approaches applied, these functions are represented within the WARSIM models by attributes and methods. We have attempted to create abridged representations of both the WARSIM Equipment and Unit models so that a visual comparison could easily be made. The following sections highlight some of the differences between the WARSIM and OMSC object models.

Platform Model Crosswalk

There appears to be about an 85 percent or better correspondence between the two object models. The WARSIM Equipment Model contains all the components of the OMSC standard except for the Logistics and Maintenance classes. The WARSIM Equipment Model represents logistics and maintenance as attributes and methods. In addition, the WARSIM Equipment Model contains a Simulated Physical Thing class. The WARSIM Team developed this abstract class as a way of capturing the operations and attributes for any simulated entity on the battlefield that has a state and is subject to detection and attrition. Figure 2 and Table 1 are provided for visual comparison between the two models.

Unit Model Crosswalk

As previously stated, the WARSIM team avoided developing class structures based on functionality. This fundamental difference in the OOA approach made the comparative crosswalk difficult. Figure 3 and Table 2 show the correspondence between the OMSC and WARSIM unit models. About 20 percent or less of the items are the same for each unit model. However, all OMSC unit model items are represented within the WARSIM unit model. The most notable differences are that the Equipment model takes care of attrition and the WARSIM C2 processes shown in Figures 4 and 5. Table 3 provides some definitions for the WARSIM classes. The below sections provide specific comments on the OMSC unit model.

Unit Class:

There is some concern over the use of the term "sides". This may inadvertently force us into the traditional red Vs blue way of thinking. Conversely, in the WARSIM model an attribute of alliance has been created to more accurately depict the real-world (we for alliances based upon common interests and goals). It appears that posture is a term used for simulation convenience for abstracting mission and Unit State. There is nothing in doctrine corresponding to posture. A mission is a large complex data structure. If mission is expected to be an enumerated value in this model then objects are needed to describe at least a rudimentary plan. An "executeMission()" is needed. In WARSIM attrition will not be determined by Unit, rather the results of combat at the platform level (WARSIM will keep track of platform location and movement as part of a formation) will be reported to Unit as damage occurs. An assessment process in Unit will maintain unit composition and status. So the "determineAttrition" method would not be used. Also, WARSIM uses heading versus MvmtDirection.

SystemGroup Class:

Within the WARSIM simulation we may have unit instances without Systems groups. Although units are composed of systems, WARSIM will model equipment separately from their units to provide additional composibility. This is different approach from the OMSC unit model.

Geometry Class:

WARSIM uses the term formation rather than shape. Within the WARSIM object model, formation is an attribute of the Unit class. Again for composibility reasons and based on the OOA approach used, WARSIM does not have a functional class like geometry. Within WARSIM, such a class might bring about a specific implementation versus being a more general representation.

C2 Class:

WARSIM has a very detailed outline for the C2 process as illustrated in Figure 4 which can be traced to the doctrinal military decision making process. The OMSC Unit model contains only doC2.

Attrition Class:

WARSIM will use attrition methods which will be executed by equipment interactions and will be maintained as part of the Equipment model.

Logistics Class:

This is handled by AEQ_Equipment.

Communications Class:

This is handled through SMCO.

Conclusion

Although there is a good amount of similarity between the OMSC Platform model and the WARSIM Equipment model, the approaches used to develop unit object models are fundamentally different. This is not to say that one approach is better than the other, rather, the WARSIM focus on satisfying training requirement and the JSIMS Enterprise influence have driven the development of WARSIM object models.

Recommendation

The WARSIM IDT has expressed interest in getting involved in the OMSC process to develop Army M&S community standards. Recommend that the OMSC contact the WARSIM IDT and possibly schedule a future meeting in Orlando. This would provide an opportunity for the WARSIM IDT to share insight into their overall development process and the thought behind their current object models.

Requirements Development Flow

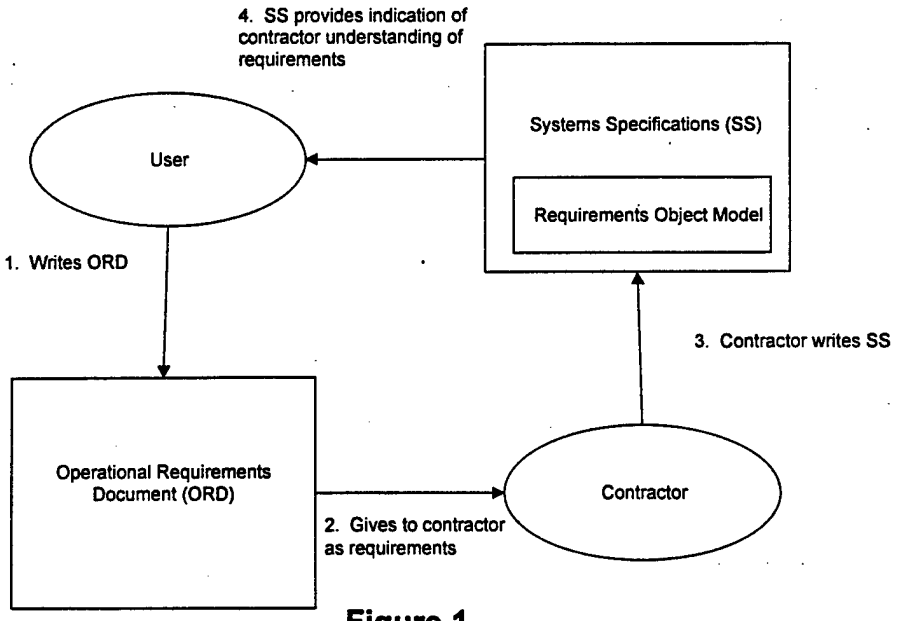
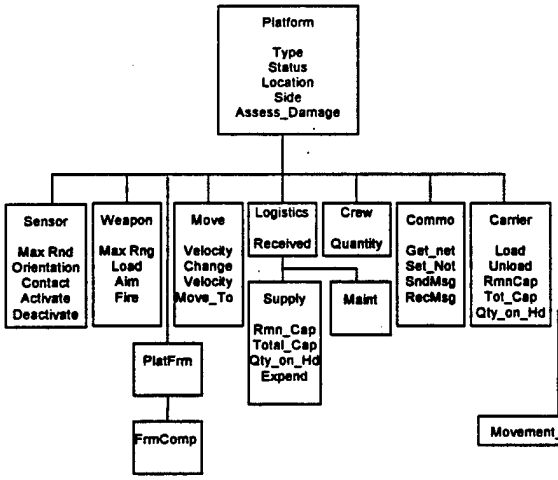


Figure 1

OMSC Platform Model



WARSIM Platform Model

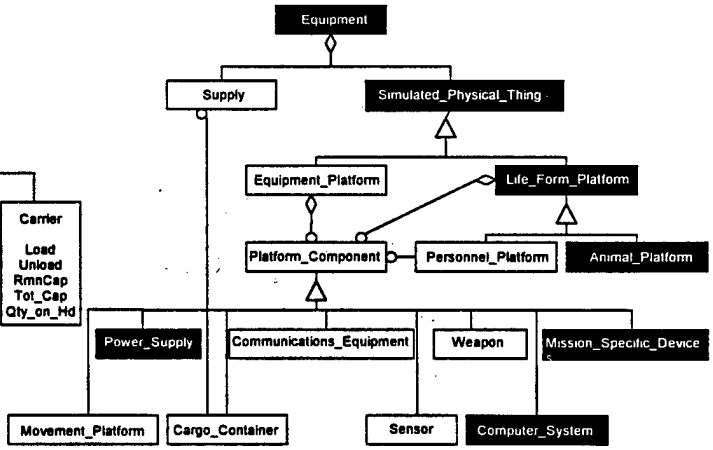


Figure 2

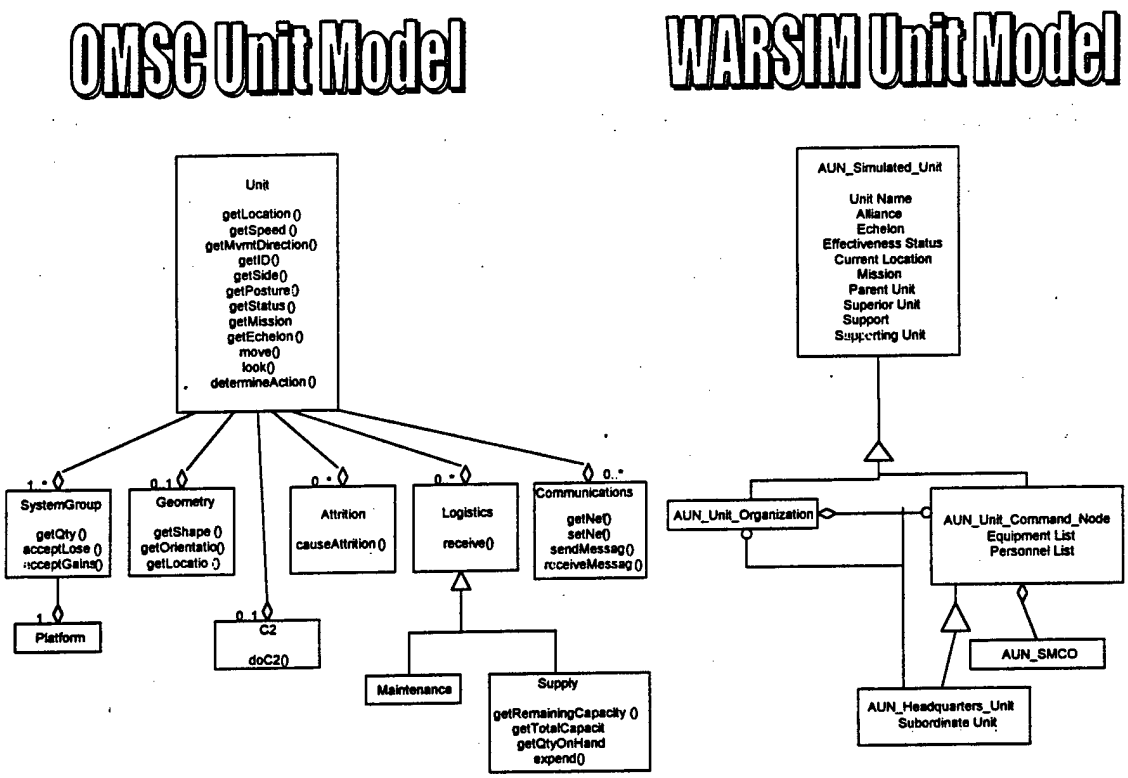


Figure 3

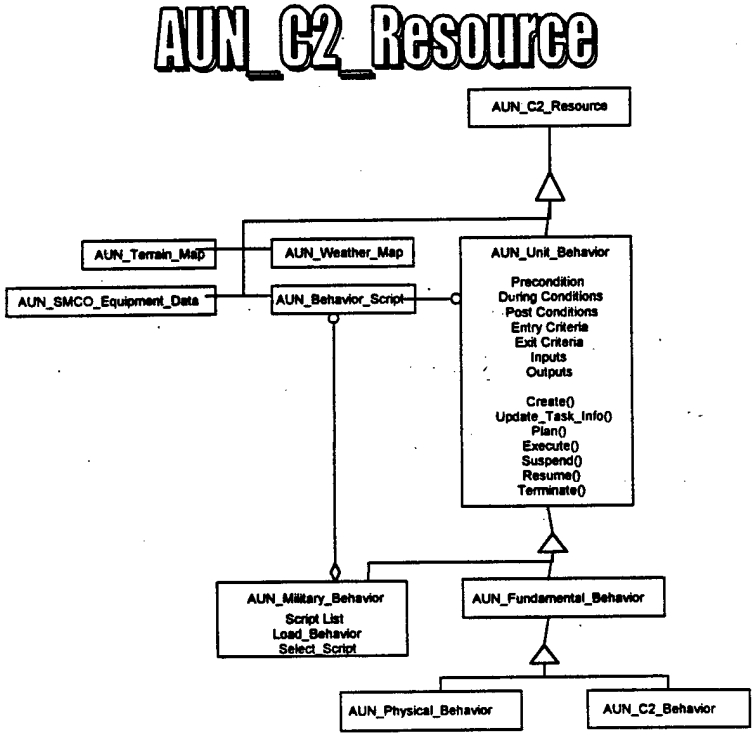


Figure 4

AUN_Unit

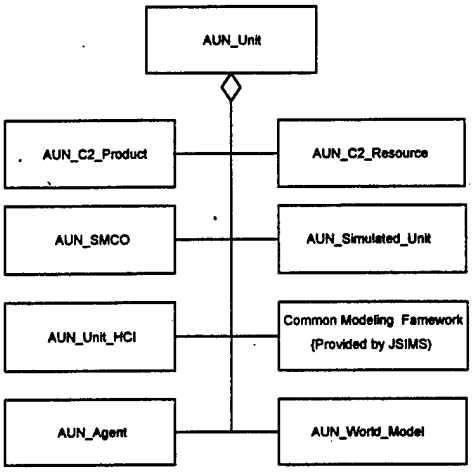


Figure 5

Table 1. Comparison of Platform Models.	
OMSC	***WARSIM***
Platform	**Equipment_Platform**
Platform Component	**Platform-Component**
Logistics Maintenance	Attributes and Methods
Supply	**Supply**
Carrier	**Cargo-Container**
Communications	**Communications-Equipment**
Crew	**Personnel-Platform**
Movement **PlatformFrame** **FrameComponent**	**Movement-Platform**
Sensor	**Sensor**
Weapon	**Weapon**

Table 2. Comparison of Unit Models.	
OMSC	***WARSIM***
Unit	**AUN_Simulated_Unit**
GetID()	Unit Name
GetSide()	Alliance
GetEchelon()	Echelon
GetStatus()	Effectiveness Status
GetLocation()	Current Location
GetMission()	Mission
GetSpeed() GetMvmtDirection() GetPosture() DetermineAction()	**AUN_C2_Behavior** (see Figure 4 for details about organization)
Move()	**AUN_Physical_Behavior** (see Figure 4 for details about organization)
Datalook()	**AUN_SMCO_Equipment_Data** passes info to **AUN_SMCO**

Table 2. Comparison of Unit Models Cont.	
OMSC	***WARSIM***
SystemGroup GetQty() AcceptLoses() AcceptGains()	**AUN_Unit_Command_Node**
Platform	**AUN_SMCO**
Geometry GetShape() GetOrientation() GetLocation()	**AUN_C2_Behavior**
C2 DoC2()	**AUN_C2_Resource**
Attrition CauseAttrition()	**AEQ_Equipment** sends info to **AUN_SMCO_Equipment_Data**
Logistics Receive()	**AEQ_Equipment**
Maintenance	**AEQ_Equipment**
Supply GetRemainingCapacity() GetTotalCapacity() GetQtyOnHand() Expend()	**AEQ_Equipment**
Communications GetNet() SetNet() SendMessage() ReceiveMessage()	**AUN_SMCO**

Table 3. Definitions.	
AEQ_Equipment	Subsystem that maintains equipment and send information about equipment to AUN_SMCO_Equipment_Data.
AUN_C2_Behavior	C2 fundamental behaviors are the atomic cognitive behaviors. The military decision making process is implemented through a combination of C2 fundamental behaviors.
AUN_Physical_Behavior	Physical fundamental behaviors have their effects in the equipment csci. All physical action of a unit occurs through physical fundamental behaviors.
AUN_Unit_Command_Node	This class represents a group of equipment and personnel at the lowest modeled echelon level that functions, and is controlled, as an atomic element. This means that the unit will behave as a single entity. For example, all of the tanks and their crews of a tank platoon will move together in a single formation.
AUN_Simulated_Unit	Unit class
AUN_SMCO	Unit command nodes have a SMCO. A unit command node's SMCO represents the minds of all the unit command node's personnel. Unit Command Node's have a specialization class called Headquarters Unit. A headquarters unit's SMCO not only directs the actions of its own physical objects, but also commands and monitors subordinate headquarters units via orders and reports.
AUN_SMCO_Equipment_Data	Contains information about the equipment.
Simulated_Physical_Thing	This object class contains the operations and attributes for any simulated entity that has a state and is subject to detection and attrition.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX C - DISTRIBUTION LIST

THIS PAGE INTENTIONALLY LEFT BLANK.

APPENDIX C - DISTRIBUTION LIST

No. of Copies	Organization
4	Director U.S. Army Model and Simulation Office ATTN: (Mr. Vern Bettencourt) (Mr. Richard Maruyama) (LTC Donald Timian) (MAJ Curt Doescher) Crystal Gateway North, Suite 503E 1111 Jefferson Davis Highway Arlington, VA 22202
1	Deputy Under Secretary of the Army for Operations Research ATTN: SAUS-OR (Mr. Walter W. Hollis) Room 2E660 102 Army Pentagon Washington, DC 20310-0102
2	Deputy Assistant Secretary for Army for Research, Development and Acquisition ATTN: SARD-ZD (Dr. Herbert Fallin, Jr.) (COL Lavine) Room 2E673 102 Army Pentagon Washington, DC 20310-0103
2	Director U.S. Army Concepts Analysis Agency ATTN: CSCA-OS (Mr. Wallace Chandler) (Mr. John Sheperd) 8120 Woodmont Avenue Bethesda, MD 20814-2797
2	Director WARSIM National Simulation Center ATTN: ATZL-NSC-W (Ms. Annette Ratzenberger) (Sean MacKinnon) 410 Kearney Avenue Fort Leavenworth, KS 66027-1306
2	Director U.S. Army TRADOC Analysis Center-FLVN ATTN: ATRC-FM (Mr. Kent Pickett) (Mr. Mike Hannon) ATTN: ATRC-TD (Dave Loental) 255 Sedgwick Avenue Fort Leavenworth, KS 66027-2345

No. of Copies	Organization
3	Director U.S. Army TRADOC Analysis Center-WSMR ATTN: ATRC-WE (Ms. Donna Vargas) (Mr. Carrol Denny) (Mr. Chad Mullis) Bldg 1401 White Sands Missile Range, NM 88002-5502
1	Commander U.S. Army TRADOC Analysis Center-Montery ATTN: MAJ Leroy Jackson PO Box 8692 Montery, CA 93940
1	Commander U.S. Army Simulation, Instrumentation, and Training Command ATTN: (Brian Saute) 12350 Research Parkway Orlando, FL 32826-3276
1	Commander U.S. Army Medical Department Center & School ATTN: MCCS-FF (Ray Devore) 1400 East Grayson Fort Sam Houston, TX 78234-6175
1	Charlie Leake JWARS Office 1555 Wilson Blvd Arlington, VA 22209
1	Tom Shook DMSTTIAC 203 Environs Road Sterling, VA 20165-5805
1	Commandant USAJFKSWCS ATTN: AOJK-DT-CD (Dean Rose) Ft Bragg, NC 28307-5000
10	Commander US Army Aviation Center ATTN: ATZQ-TDS-W (Rarick) Ft Rucker, AL 36362-5263

No. of Copies	Organization
1	CPT David Dinger Bldg. 5G, Room 303 DCSSA, HQ TRADOC Fort Monroe, VA 23651
10	Director U.S. Army Materiel Systems Analysis Activity 392 Hopkins Road ATTN: AMXSY-C AMXSY-CS (Alan Dinsmore, Brad Bradley) AMXSY-J (Pete Rigano) AMXSY-DD (3 cys) Aberdeen Proving Ground, MD 21005-5071
1	Assistant Secretary of the Army for Research, Development, and Acquisition ATTN: SARD-DO (Ms. Ellen Purdy) 2511 Jefferson Davis Highway Arlington, VA 22202-3911
1	Commander Headquarters U.S. Army Corps of Engineers Director of Research and Development ATTN: CERD-M (Mr. Jerry Lundien) 20 Massachusetts Avenue, NW Washington, DC 20312-1000
1	Commander U.S. Army Opertional Test and Evaluation Command ATTN: CSTE-M (Ms. Sarah Wilson) 4501 Ford Avenue Alexandria, VA 22302-1458
1	Director U.S. Army Cost and Economic Analysis Center ATTN: SFFM-CA-PA (Mr. Steve Pawlow) 5611 Columbia Pike Falls Church, VA 22041-5050
1	Director U.S. Army Concepts Analysis Agency ATTN: CSCA-OS (Mr. Gerry Cooper) 8120 Woodmont Avenue Bethesda, MD 20814-2797

No. of Copies	Organization
2	Commander, DCSSA US Army Training and Doctrine Command ATM: ATAN-SM (Mr. Carson/Angela Winter) Fort Monroe, VA 23651-5143
1	Commander, U.S. Army Material Command ATTN: AMCRDA-TL (Mr. Ken Welker) 5001 Eisenhower Ave Alexandria, VA 22333-0001
1	Commander US Army Space and Missile Defense Command ATTN: CSSD-BL-SC (Mr. Troy, Street) PO Box 1500 Huntsville, AL 35807
1	Commandant, US Army War College ATTN: AWC-AW (COL Pat Slattery) Carlisle Barracks Carlisle, PA 17013-5050
1	Chief of Army Reserves ATTN: DAAR-PAE (CPT Ward Litzenberg) Room ID416, Pentagon Washington, DC 20310-2400
1	Director U.S. Army Logistics Integration Agency ATTN: LOSA-CD (Mr. Mike Rybacki) 54 M Avenue, Suite 4 New Cumberland, PA 17070-5007
1	Commander U.S. Army Signal Command ATTN: AFSC-PLE-AM (Dr. Leon Spencer) Fort Huachuca, AZ 85613-5000
1	Commander U.S Army Forces Command ATTN: AFOP-PLA (MAJ Steve Aviles) 1777 Hardee Avenue, S.W. Fort McPherson, GA 30330-6000
1	Deputy Chief of Staff for Intelligence ATTN: DAMI-IFT(Ms Marilyn Macklin) Room 9302, Presidential Tower 2511 Jefferson Davis Highway Arlington, VA 22202

No. of Copies	Organization
1	Commander U.S. Army Research Institute for the Behavioral and Social Sciences ATTN: PERI-II (Dr. Philip Gillis) 12350 Research Parkway Orlando, FL 32826
1	Office of the Chief of Staff, Army Progam Analysis & Evaluation Directorate ATTN: DACS-DPM (LTC Mike Clark) Room 3C719, Pentagon Washington, DC 20310
1	Deputy Chief of Staff for Personnel ATTN: DAPE-MR (Dr. Robert Holz) Room 2C733, Pentagon Washington, DC 20310
1	Chief, National Guard Bureau ATTN: NGB-ARO-TS (MAJ Gary Harber) 111 South George Mason Drive Arlington, VA 22204-1382
1	Army Digitization Office ATTN: DAMO-ADO (Ms. Susan Wright) 400 Army, Pentagon Washington, DC 20301
1	Military Traffic Management Command Tansportation Engineering Agency (MTMCTEA) ATM: MTTE-SIM (Mr. Melvin Sutton) 720 Thimble Shoals Boulevard, Suite 130 Newport News, VA 23606
1	Commander-in-Chief U.S. Army Europe and 7th Army ATTN: AEAGC-TS-F (LTC Howard Lee) Unit: 28130 APO AE 09114
1	Commander HQ, USARPAC ATTN: APOP-PL (Mr. Bob Deryke) Foprt Shafter, HI 96858-5100
1	TRAC-WSMR ATTN: ATRC-WB (Dave Dixon) WSMR, NM 88002-5502

No. of Copies	Organization
1	Commander USASTRICOM ATTN: AMSTI-EC (Wesley Milks) 12350 Research Parkway Orlando, FL 32826-3276
	Commander USA Signal Center and Fort Gordon ATTN: DCD. CAD, M&S Br (Burt Kunkel) Fort Gordon, GA 30905-5090
1	Director USA Research Laboratory ATTN: AMSRL-IS-EW (Rick Shirkley) WSMR, NM 88002-5501
1	PM-WARSIM USA STRICOM ATTN: MAJ Frank Rhinesmith 12350 Research Parkway Orlando, FL 32826
1	USA CASCOM ATTN: ATCL-CAT (Ron Fischer) Fort Lee, VA 23801-6000
1	Director USACAA ATTN: CSCA-MD (Julie Allison) 8120 Woodmont Avenue Bethesda, MD 20814-2797
1	Director USAEWES ATTN: CEWES-GM-K (Niki Deliman) 3909 Halls Ferry Road Vicksburg, MS 39181-6199
1	TRAC ATTN: ATRC-FM (Pam Blechinger) 255 Sedgewick Avenue Fort Leavenworth, KS 66027-1306
1	USATEC ATTN: CETEC-TP (Ken Barnette) 7701 Telegraph Road Alexandria, VA 22315

No. of Copies	Organization
1	TPIO for Synthetic Environment National Simulation Center ATTN: MAJ Mike Staver 410 Kearney Avenue' Fort Leavenworth, KS 66027-1306
7	Defense Technical Information Center 8725 John J. Kingman Road, STE 0944 Fort Belvoir, VA 22060-6218
1	TRAC ATTN: ATRC-FZ (Larry Cantwell) 255 Sedgwick Avenue Fort Leavenworth, KS 66027-2345
1	Director Information Systems for Command, Control, Communications, & Computers ATTN: SAIS-PAA-S (LTC Craig Cromwell) Room 1C634, Pentagon Washington, DC 20310
1	Deputy Chief of Staff for Operations and Plans ATTN: DAMO-ZD (MAJ Bruce Simpson) Room 3A538, Pentagon Washington, DC 20310-0400